

del  
61  
1. (Currently Amended) A method of factoring operating system functions comprising:

defining criteria that governs how functions of an operating system are to be factored into one or more groups;

factoring the functions into one or more groups based upon the criteria; and  
associating ~~groups~~ of functions with programming objects that have data and methods, wherein the methods correspond to the operating system functions effective to provide an object oriented operating system, the programming objects being configured to be instantiated throughout a remote computing system.

2. (Original) The method of claim 1, wherein the programming objects have interfaces through which the methods can be accessed.

3. (Original) The method of claim 1, wherein the programming objects comprise COM objects.

4. (Original) The method of claim 1, wherein said factoring comprises creating a hierarchy of object interfaces in which certain interfaces can inherit from other interfaces.

5. (Original) The method of claim 1, wherein said factoring comprises creating a hierarchy of object interfaces in which certain interfaces can aggregate with other interfaces.

1 6. (Original) The method of claim 1 further comprising instantiating a  
2 plurality of programming objects across a process boundary.

3  
4 7. (Original) The method of claim 1, further comprising instantiating a  
5 plurality of programming objects across a machine boundary.

6  
7 8. (Original) The method of claim 1, wherein the criteria is based, at  
8 least in part, on the manner in which particular functions behave.

9  
10 9. (Original) The method of claim 8, wherein the manner includes a  
11 consideration of the types of operating system resources that are associated with  
12 the operation of a function.

13  
14 10. (Original) The method of claim 8, wherein the manner includes a  
15 consideration of whether a particular function creates an operating system  
16 resource.

17  
18 11. (Original) The method of claim 8, wherein the manner includes a  
19 consideration of whether a particular function operates upon an operating system  
20 resource.

21  
22 12. (Original) The method of claim 1, wherein the criteria is based, at  
23 least in part, on the manner in which particular functions behave, wherein the  
24 manner includes:  
25

1 a consideration of the types of operating system resources that are  
2 associated with the operation of a function; and

3 a consideration of whether a particular function creates an operating system  
4 resource.

5  
6 13. (Original) The method of claim 1, wherein the criteria is based, at  
7 least in part, on the manner in which particular functions behave, wherein the  
8 manner includes:

9 a consideration of the types of operating system resources that are  
10 associated with the operation of a function call; and

11 a consideration of whether a particular function operates upon a given  
12 operating system resource.

13  
14 14. (Currently Amended) A method of factoring operating system  
15 functions comprising:

16 factoring a plurality of operating system functions that are used in  
17 connection with operating system resources into first groups based upon first  
18 criteria;

19 factoring the first groups into individual sub-groups based upon second  
20 criteria; and

21 assigning each sub-group to its own programming object interface, wherein  
22 a programming object interface represents a particular object's implementation of  
23 its collective methods effective to provide an object-oriented operating system,  
24 wherein individual objects having associated programming object interfaces are  
25 configured to be instantiated throughout a remote computing system.

1  
2 15. (Original) The method of claim 14, wherein the first criteria is based  
3 upon the type of resource that is associated with an operation of a function.

4  
5 16. (Original) The method of claim 14, wherein the second criteria is  
6 based upon the nature of an operation of a function on a particular resource.

7  
8 17. (Original) The method of claim 16, wherein said nature concerns  
9 whether a function creates a resource.

10  
11 18. (Original) The method of claim 16, wherein said nature concerns  
12 whether a function does not create a resource.

13  
14 19. (Original) The method of claim 14, wherein the first criteria is based  
15 upon the type of resource that is associated with an operation of a function, and the  
16 second criteria is based upon the nature of an operation of a function on a  
17 particular resource.

18  
19 20. (Original) The method of claim 14, wherein at least one interface  
20 inherits from another interface.

21  
22 21. (Original) The method of claim 14, wherein at least one interface  
23 aggregates with another interface.

1 22. (Original) The method of claim 14 further comprising instantiating a  
2 plurality of programming objects across a process boundary.

3  
4 23. (Original) The method of claim 14 further comprising instantiating a  
5 plurality of programming objects across a process boundary and a machine  
6 boundary.

7  
8 24. (Currently Amended) A method of factoring operating system  
9 functions comprising:

10 factoring a plurality of operating system functions into interface groups  
11 based upon the resources with which a function is associated;

12 factoring the interface groups into interface sub-groups based upon each  
13 function's use of a handle that represents a resource; and

14 organizing the interface sub-groups so that at least one of the interface sub-  
15 groups inherits from at least one other of the interface sub-groups, individual  
16 interface sub-groups being associated with individual programming objects that  
17 can be instantiated throughout a remote computing system.

18  
19 25. (Original) The method of claim 24, wherein said organizing  
20 comprises aggregating at least one of the interface sub-groups.

21  
22 26. (Original) The method of claim 24, wherein the interface sub-groups  
23 are associated with COM objects.  
24  
25

1 27. (Original) The method of claim 24, wherein the factoring of the  
2 interface groups into interface sub-groups comprises considering whether a  
3 function creates a handle.  
4

5 28. (Original) The method of claim 24, wherein said organizing  
6 comprises aggregating at least one of the interface sub-groups, and wherein the  
7 factoring of the interface groups into interface sub-groups comprises considering  
8 whether a function call creates a handle.  
9

10 29. (Previously Amended) An operating system application program  
11 interface embodied on a computer-readable medium comprising a plurality of  
12 object interfaces, wherein each object interface is associated with an object that  
13 includes one or more methods that are associated with and can call functions of an  
14 operating system that does not comprise the object interfaces, individual objects  
15 being configured to be instantiated in process, locally, or remotely.  
16

17 30. (Original) The operating system application program interface of  
18 claim 29, wherein the object interfaces are arranged in groups in accordance with  
19 the types of objects with which their operation is associated.  
20

21 31. (Original) The operating system application program interface of  
22 claim 29, wherein the methods within some of the interfaces are arranged in  
23 accordance with whether they create an object.  
24  
25

1 32. (Original) The operating system application program interface of  
2 claim 29, wherein the methods within some of the interfaces are arranged in  
3 accordance with whether they do not create an object.  
4

5 33. (Original) The operating system application program interface of  
6 claim 29, wherein the methods within some of the interfaces are arranged in  
7 accordance with whether they operate upon an object.  
8

9 34. (Original) The operating system application program interface of  
10 claim 29, wherein at least some of the object interfaces are arranged so that they  
11 inherit from other of the object interfaces.  
12

13 35. (Original) The operating system application program interface of  
14 claim 29, wherein at least some of the object interfaces are arranged so that they  
15 aggregate with other of the object interfaces.  
16

17 36. (Currently Amended) An operating system comprising:  
18 a plurality of programming objects having interfaces, wherein the  
19 programming objects represent operating system resources, and wherein the  
20 interfaces define methods that are organized in accordance with whether they  
21 create an operating system resource or not;

22 wherein the programming objects are configured to be called either directly  
23 or indirectly by an application; and

24 wherein the methods are configured to call operating system functions  
25 responsive to being called directly or indirectly by an application;

1 said programming objects being configured to be instantiated throughout a  
2 remote computing system.

3  
4 37. (Original) The operating system of claim 36, wherein some of the  
5 objects are disposed across at least one process boundary.

6  
7 38. (Original) The operating system claim 36, wherein some of the  
8 objects are disposed across at least one machine boundary.

9  
10 39. (Previously Amended) The operating system of claim 36, wherein at  
11 least some of the objects are disposed across at least one process boundary and at  
12 least one machine boundary.

13  
14 40. (Previously Amended) The operating system of claim 36, wherein  
15 the objects comprise COM objects.

16  
17 41. (Currently Amended) A method of converting an operating system  
18 from a non-object-oriented format to an object oriented format, wherein the  
19 operating system includes a plurality of operating system functions that are  
20 callable to create or use operating system resources, the method comprising:

21 defining a plurality of programming object interfaces that define methods  
22 that correspond to the operating system functions, wherein programming objects  
23 that support the interfaces are callable either directly by an application that makes  
24 object-oriented calls, or indirectly by an application that makes function calls, said  
25



1 programming objects being configured to be instantiated throughout a remote  
2 computing system;

3 calling a programming object interface either directly via an object-oriented  
4 call, or indirectly via an indirection that transforms a function call into an object-  
5 oriented call; and

6 responsive to said calling, calling an operating system function with a  
7 method of the programming object that supports said programming object  
8 interface.